IMPROVED METHODS FOR FAST SYSTEM RELIABILITY ANALYSIS THROUGH MACHINE-LEARNING-BASED SURROGATE MODELS

BY

RAPHAEL EPHRAIM STERN

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Civil Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2015

Urbana, Illinois

Master's Committee:

Assistant Professor Daniel B. Work
Associate Professor Junho Song, Seoul National University

ABSTRACT

In the aftermath of a natural disaster, knowledge of the connectivity of different regions of infrastructure networks is crucial to post-event decision making. The specific problem of determining the probability that two nodes in an infrastructure network are disconnected given the edge failure probabilities is known as the two-terminal connectivity problem, a special case of the k-terminal reliability problem. Both problems are known to be computationally intractable for general infrastructure graphs as the network size grows large, which motivates the use of Monte Carlo techniques to estimate the failure probability. However, Monte Carlo techniques are slow to converge due to the large number of realizations of the infrastructure graph required, each of which requires a connectivity evaluation. To improve the computation efficiency of the Monte Carlo approach, this work develops a new framework where the connectivity evaluation is itself estimated with a machine-learning-based surrogate model.

The framework is applied to networks with both uncorrelated uniform edge failure probability and correlated edge failure probability, and an extension to node clusters is also proposed. The method first uses spectral clustering to partition the network, and estimates the connectivity of these clusters using both a logistic regression and an AdaBoost classifier.

Numerical experiments on a California gas distribution network demonstrate that using the surrogate model to determine cluster connectivity in-

troduces less than five percent error and is two orders of magnitude faster than methods using an exact connectivity evaluation to estimate the probability of network failure through Monte Carlo simulations.

To David, for always keeping a positive outlook

ACKNOWLEDGMENTS

I would like to thank my advisers Dr. Daniel Work and Dr. Junho Song for their help and support throughout my time so far at the University of Illinois. You showed me that there are interesting problems to be solved everywhere you look, and that some problems are far more complex than they seem at the surface. Your suggestions and comments challenged me, and helped me become a better researcher. I would also like to thank the entire Work Lab in 3117 for all the fun times we've had together. Whether it was in the lab, at a conference, or at the Pig, with you, research was fun.

I would also like to thank my parents, Louis and Franziska Stern for all they have done for me to bring me to this point. You taught me so much. Your love and support has made this possible.

Finally, I would like to thank my siblings, Hannah Stern, Immanuel Stern, and Gabriel Stern, as well as Hannah Burson. The time we have spent together talking, traveling, hiking, and just having fun has meant so much to me. Each of you have provided more pride and inspiration to me than you can ever imagine.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

COV	Coefficient of Variation
DFS	Depth First Search
DHS	Department of Homeland Security
FEMA	Federal Emergency Management Agency
MCS	Monte Carlo Simulation
ML	Machine Learning
PGV	Peak Ground Velocity
SPG	Spectral Graph Clustering
SRA	System Reliability Analysis

CHAPTER 1

INTRODUCTION

1.1 Motivation and problem statement

In the aftermath of a natural disaster, prompt response is critical to minimize economic damage and the loss of life [1, 2]. In order to improve post–disaster response times to hazards that impact gas, water, and electricity distribution systems, as well as other lifeline networks, efficient methods are needed to quickly and accurately estimate the network failure probability under a given set of failure probabilities of individual components. *System reliability analysis* (SRA) of infrastructure networks encompasses a number of methods to determine the probability that a network will be able to complete its designed function after a disaster event. Such methods are used to analyze the resilience of the network with respect to failure under some disaster event [3, 4, 5, 6].

In recent years, resilience has become a central component to urban design. Resilient cities have gathered increasing research attention, particularly following several high-impact natural disasters. The *Federal Emergency Management Agency* (FEMA) and the *Department of Homeland Security* (DHS), as well as the Office of the President of the United States have repeatedly stressed the importance of resilient infrastructure in recent years [7, 8]. These reports have gone so far as to state that the well-being of our nation relies on secure and resilient infrastructure.

For the purposes of SRA, such infrastructure networks can be modeled as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the vertex (node) set, $\mathcal{E} \subseteq \{(i,j) : i,j \in \mathcal{V}\}$ is the set of edges, and the number of edges $n = |\mathcal{E}|$ is the cardinality of the edge set. This network can considered to be a *stochastic network* since each edge fails with a certain probability, p_f.

In such stochastic networks, the exact probability $\Pr(fail)$ that two specific network nodes have been disconnected following a disruptive event is a quantity of interest to emergency management personnel. The objective of this thesis is to explore and develop methods to quickly approximate this probability as $\hat{\Pr}(fail)$.

1.2 Related work

The problem of interest is thus a subset of the *n-terminal reliability* problem, and specifically, the *two-terminal reliability* problem [9]. This problem has been studied extensively for infrastructure and communications networks [10, 11, 12]. In general, this problem has been shown to be $\sharp\mathcal{P}$-*complete*, which is a family of counting problems that are at least as difficult to solve as $\mathcal{NP}$-*complete* problems [13, 14]. Thus, it is unlikely that there exists a polynomial-time solution to solve the exact two-terminal reliability problem. However, it is worth noting that there exists a subset of graphs for which the two-terminal reliability problem can be solved efficiently. For graphs consisting of series-parallel subgraphs, as well as graphs with a bounded path width, decomposition algorithms exist to solve reliability problems in linear time [15]. Unfortunately, most infrastructure networks do not fall into these categories, and thus, exact methods require an exponential number of enumerations which is not computationally tractable for large networks[15].

Another approach is to estimate the probability of disconnection. This can be done using *Monte Carlo simulation* (MCS). For example, Karger [16] uses (MCS) to compute an approximate failure probability and provides bounds on the number of simulations required to achieve a converged estimate within a specific error bound. For infrastructure failure analysis, Hwang, et al. [17] uses MCS to analyse the seismic resilience of the Shelby, Co. Tennessee water delivery network, the post-disaster network is analyzed for only 50 runs of the MCS due to the computational complexity of the problem. Further, Karger [16] shows that MCS is appropriate when the individual component failure probabilities are greater than n^{-4}.

In general, MCS works by simulating many network states with differing combinations of failed edges. For each realization of the MCS, the network is *damaged*, and connectivity of the *source* and *terminal* nodes, s and t, respectively, are checked. Rubino and Tuffin [18] use *depth first search* (DFS) to determine the connectivity of s and t. As a result, each connectivity evaluation is computed in $O(|\mathcal{V}|)$ time [19]. Surrogate models have been introduced in many cases when exact computation is not technically feasible [20, 21]. The question this thesis address is how to construct an accurate surrogate model to be used to determine graph node connectivity for each network realization when conducting MCS.

1.3 Contributions and outline of this thesis

The primary contribution of this thesis is to propose and implement a framework to quickly estimate the probability of network failure using MCS, and to demonstrate two orders of magnitude reduction in computation time for approximating the probability of network failure. The work presented describes

how to construct a machine-learning-based surrogate model for network connectivity. Extensions to connectivity of node clusters as well as applications of this work to the California gas distribution network with realistic, correlated, edge failures are also discussed.

The remainder of the thesis is organized as follows. In Chapter 2 a general framework for approximate SRA is described, as well as the methods used to construct a machine-learning-based classifier. Application of these techniques to a simplified version of the California gas distribution network is demonstrated in Chapter 3, and demonstrates a reduction in computation time over traditional methods. Finally, in Chapter 4, the conclusions show that the use of a machine-learning-based surrogate model for network connectivity can be used to replace exact connectivity methods to efficiently estimate the probability of network failure.

CHAPTER 2

DEVELOPING APPROXIMATE SYSTEM RELIABILITY ANALYSIS

2.1 Monte Carlo framework

To conduct MCS for network reliability, the network and component states are first defined. Using an infrastructure graph $\mathcal{G}$, the network state is defined as $x \in \{0,1\}^n$, where each element x_i encodes the state of the corresponding network component (edge) as

$$x_i = \begin{cases} 0 & \text{if edge } i \text{ has failed,} \\ 1 & \text{if edge } i \text{ remains intact.} \end{cases} \tag{2.1}$$

Similarly, the network status $y_{s,t} = f(x)$ with respect to source node s and terminal node t is a function of the network state and is defined as follows:

$$y_{s,t} = \begin{cases} 1 & \text{if } s \text{ and } t \text{ are connected,} \\ -1 & \text{otherwise,} \end{cases} \tag{2.2}$$

where connectivity is defined as the existence of a path between the two nodes.

When using MCS methods to estimate the probability of network failure, the network state is simulated for K realizations, and the network status for the k^{th} realization, $y_{s,t}(k)$ is determined by allowing edges to fail randomly with some probability p_f, and removing these failed edges from the graph

before determining connectivity of s and t. Based on the findings of Karger [16], K is selected such that

$$K = O\left(\frac{\log n}{\epsilon^2 \Pr(fail)}\right), \tag{2.3}$$

where $\Pr(fail)$ is the probability of network failure, to obtain an estimate within $1 \pm \epsilon$. Generally, $\Pr(fail)$ is defined as:

$$\Pr(fail) = \Pr(s, t \text{ disconnected} | \mathcal{G}, p_f). \tag{2.4}$$

To evaluate $\hat{\Pr}(fail)$, the estimate on the probability that no path exists between nodes s and t – the network failure probability – is computed as

$$\hat{\Pr}(fail) \approx \frac{1}{K} \sum_{k=1}^{K} y_{s,t}(k). \tag{2.5}$$

This general framework is outlined in Figure 2.1 where a *hazard model* is used to determine p_f for each edge following some disaster event, E. These edge failure probabilities are then used to generate network realizations x, which are used to estimate the network failure probability, $\hat{\Pr}(fail)$ using MCS.

Finally, the *coefficient of variation* (COV) of the network failure probability estimate $\delta_{\hat{\Pr}(fail)}$ is computed as

$$\delta_{\hat{\Pr}(fail)} = \frac{s_N}{\sqrt{K} \cdot \hat{\Pr}(fail)}. \tag{2.6}$$

Here s_N is the sample standard deviation of network failure probability estimates, and K is the total number of MCS. This is used to determine how much each additional trial is influencing the estimated value, and provides

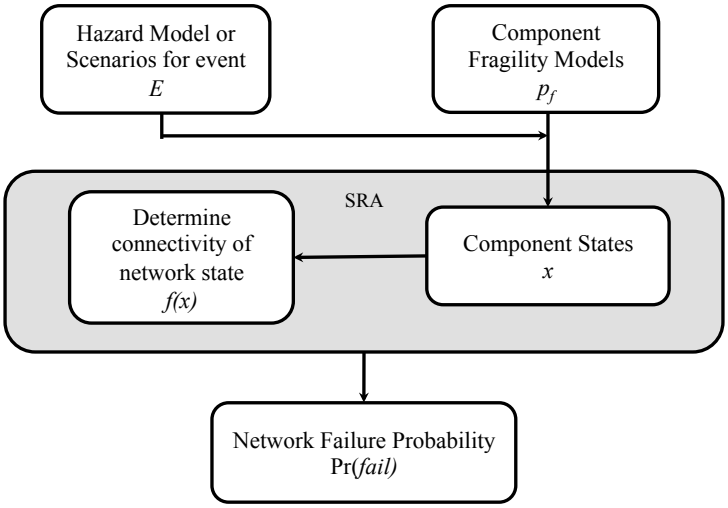


Figure 2.1: General MCS framework for computing the probability of network failure.

insight into the confidence that the estimate has converged.

2.2 Exact vertex connectivity

As demonstrated by Hummon and Doreian, as well as Even and Tarjan,[22, 23], *depth first search* (DFS) can be used to check connectivity of two nodes in each MCS trial. In graphical analysis, DFS is an algorithm that is commonly used to quickly traverse a graph. While DFS does not provide a guaranteed shortest path between the source and terminal node, it can be used check connectivity since it provides an upper bound on the shortest path length. If this path length is not defined, then the nodes are not connected through the network. The computation time associated with traversing the graph is $O(|\mathcal{E}|)$. This is significantly faster than shortest path algorithms such as

Dijkstra's algorithm [24], which requires $O(|\mathcal{V}|^2)$ to compute the shortest path between any two nodes. This makes DFS a natural choice for quickly checking connectivity between two nodes.

Depth first search is a recursive algorithm that begins at a source node, and explores as far as possible along each path before backtracking to the last node with unvisited neighbors when the end of a path is reached. Upon backtracking, the algorithm recursively follows links to neighboring nodes. This is continued until all possible nodes have been reached. The algorithm maintains a set of nodes which have been visited. If, upon completion, the terminal node is not in the set of visited nodes, the source and terminal nodes have been disconnected. The full DFS algorithm is outlined in Algorithm 1 [25] which uses a stack to track the nodes to be visited. G is the graph to perform the search on, and v is the starting node.

Algorithm 1 Depth First Search

```
DFS(G, v) (v is the vertex where the search starts)
Stack S := ; (start with an empty stack)
for each vertex u do
    set visited[u] := false; push S, v;
    while (S is not empty) do
        u := pop S;
        if (not visited[u]) then
            visited[u] := true;
            for each unvisited neighbor w of u do
            push S, w;
            end for
        end if
    end while
end for
```

2.3 Surrogate models for vertex connectivity

Surrogate models are commonly used to simplify computations in fields as diverse as structural optimization, waste water modeling, and supply chain management [26, 27, 28]. Generally, a surrogate model is one that allows a complex system to be simply modeled. More formally, a surrogate model is a function that approximates some function $f(\cdot)$ with some simpler function $\tilde{f}(\cdot)$ such that

$$\tilde{f}(\cdot) \approx f(\cdot) \tag{2.7}$$

holds.

In complex, infrastructure-scale, networks even very fast methods such as DFS become too time consuming. For such networks the idea of constructing a surrogate model for network connectivity by approximating network behavior with a machine-learning based surrogate model is introduced. This allows for quickly determining whether two nodes are connected, but introduces uncertainty due to the imprecise nature of surrogate models. To construct such a surrogate model, a supervised learning method is used.

The resulting framework for such SRA is shown in Figure 2.2. Here, training data is used to construct a *machine learning* (ML) classifier in the model training step. This classifier is used for connectivity determination in the network SRA step where a hazard model is used to determine component failure probabilities which are used to sample network realizations.

An approach to estimate whether two nodes in the network are connected through a supervised machine-learning based classifier using *logistic regression* is developed subsequently in Section 2.4. The application of this method occurs in two steps: training and application. To train the model a representative set of training data is required for the model to learn the network

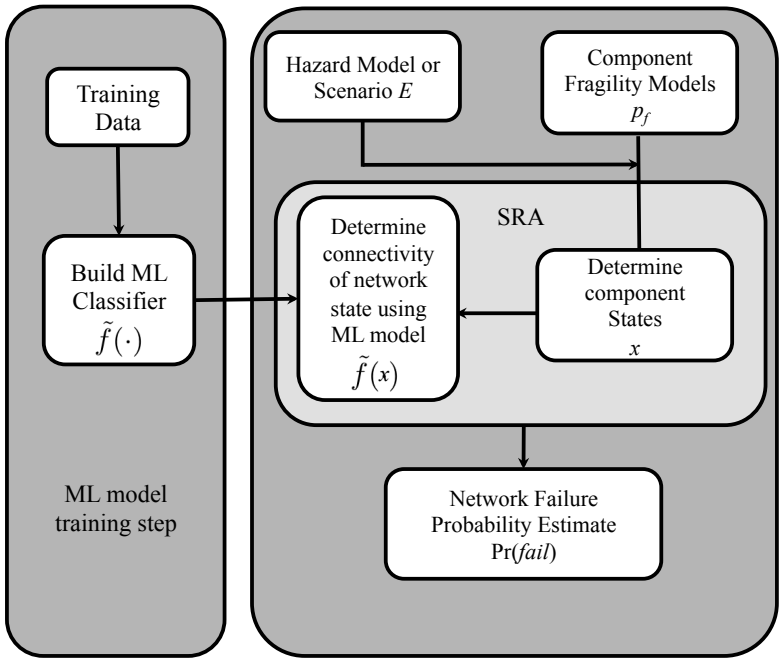


Figure 2.2: Framework for approximate SRA using MCS with surrogate model.

behavior. The use of random sampling is shown to produce training sets that are suitable for logistic regression.

2.4 Logistic regression for surrogate model

To train the surrogate model, a representative set of training data is required for the model to learn the network behavior. Logistic regression is a probabilistic statistical classification model that is used to predict a binary response from a binary predictor. It is often used to predict the probability of failure since it makes a classification decision based on a threshold which mimics physical failure [29]. Both network and component failure are binary, meaning that they either fail or remain intact. Instead of providing a *hard* classification like many binary classification methods, logistic regression pre-

dicts the probability that a data point belongs to each class. However, it can be used as a hard classifier by applying a probability threshold. This makes logistic regression a relevant method for a surrogate model for network failure with binary component failure states as the features.

The probability that datapoint x belongs to the default class is computed as

$$F(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1^T x)}}, \tag{2.8}$$

where $F(x)$ gives the probability of disconnection of network realization x, and β_0 is a scalar that represents an offset, and β_1 is the vector of model parameters for each feature, both of which are learned from training data. The decision threshold is set at $F(x) = 0.5$ to provide a hard classification rule. The regression coefficients can be learned using *maximum likelihood* estimation. This is typically done using numerical methods such as Newton's method [30], since it is not possible to find a closed-form expression for the coefficients.

Intuitively, the values of β_1 can be thought of as relative importance factors for each element in the network. This indicates that some elements, which have a higher weight, are relatively more important to network survival than others with lower weight.

2.5 Boosting-based surrogate models

Alternately to a logistic regression-based surrogate model, a *boosting*-based classifier is also considered. The methodology for constructing such a surrogate model for network failure is outlined in this section.

In machine learning, boosting is a method that aggregates several *weak* predictions to construct one more accurate prediction [31]. Specifically, the

AdaBoost algorithm, short for "Adaptive Boosting", is a classifier that combines multiple weighted weak classifiers that, when combined, produce a sophisticated classifier. AdaBoost is selected since it has been shown to learn complex structure with limited training data [32].

The final classifier is given as:

$$\tilde{f}(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right), \tag{2.9}$$

which is a linear combination of T weak classifiers h_t with weights α_t.

Each weak classifier h_t is a one dimensional classification rule. AdaBoost sequentially computes each weak classifier, indexed by t, and determines its contribution α_t to the overall classifier according to:

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right) > 0, \tag{2.10}$$

where ϵ_t is the classification error of the t^{th} weak classifier, defined subsequently.

The weak classifier h_t is determined by minimizing the classification error of the m training data points $\{(x(1), y(1)), \ldots, (x(m), y(m))\}$ as:

$$\epsilon_t = \frac{1}{m}\sum_{k=1}^{m} W_t(k)\left(h_t(x(k)) \neq y(k)\right). \tag{2.11}$$

To compute the classification error, the datapoints are weighted using (2.12).

$$\{W_t(1), \cdots, W_t(m)\} \tag{2.12}$$

For the first classifier, these weights are initialized as $W_1(k) = 1/m$. For subsequent classifiers, the new weight for datapoint $x(k)$ is computed from

the previous data weight $W_{t-1}(k)$, the previous classifier h_{t-1} operating on $x(k)$, and the classifier weight α_{t-1} as follows:

$$W_t(k) = \frac{W_{t-1}(k)}{Z_{t-1}} \exp\left(-\alpha_{t-1} y(k) h_{t-1}\left(x(k)\right)\right), \tag{2.13}$$

where Z_t is a normalization constant.

2.6 Sampling methods

The quality of predictions are only as good as the data used to train the classifier. Since the size of the state space is exponentially proportional to the size of the network (i.e., 2^n possible states), even modestly-sized infrastructure networks have an intractably large number of states to consider. Therefore, a subset of these states is used to train the classifier.

This section discusses two methods for sampling data, each produces training data suitable for a different machine learning method as will be shown in Chapter 3. The randomly generated data is acceptable for logistic regression, while AdaBoost requires a more balanced dataset. Thus a random walk sampling technique is developed.

2.6.1 Sampling data for logistic regressions

Due to the relative robustness of logistic regression to imbalanced training data, the training data used for this method employs a uniform random distribution on the number of edges which fail. For each datapoint, first the number of edges that fail is selected from a uniform distribution. This number is selected to range from 0 to n, the number of edges in the network. Next, the individual edges to fail are selected uniformly at random, and the

network is damaged correspondingly. Connectivity of s and t is checked using DFS described in Section 2.2, and the result is recorded as the training label.

2.6.2 Sampling data for boosting

For best results, AdaBoost is trained on data that has an equal number of connected and disconnected states [33]. However, due to the the inherent robustness of infrastructure networks, uniform random sampling would lead to imbalanced training data. Therefore, it is important to preferentially select certain states to obtain an appropriate, balanced, training data set.

A random-walk sampling technique based on the Metropolis-Hastings algorithm [34] to select points for a training dataset is proposed. Network states are categorized based on the number of failed edges, n_f. The proportion of observed network states with n_f edge failures that correspond to failed networks, p_{n_f}, is tracked throughout the random walk. At each step a trial point with n_f^t edge failures, and a historic $p_{n_f}^t$ based on all previously sampled network realizations with n_f^t edge failures is proposed. To select points such that the number of connected and disconnected network states are balanced, points are sampled to minimize

$$B_k = \left| p_{n_f} - 0.5 \right| \tag{2.14}$$

by preferentially selecting trial points with

$$B_k^t = \left| p_{n_f}^t - 0.5 \right| < B_k.$$

The guided random walk sampling algorithm exists in two phases. In the first phase, the algorithm randomly selects a small number of network states

for all possible n_f (i.e., $1, 2, \ldots, n$) and determines connectivity of the source and terminal node for each of these network states. Once a baseline estimate on the probability of network failure is established, the algorithm continues to the random-walk phase. For the random walk, a trial point n_f^t is obtained from the previous point n_f by taking a Gaussian-random step which follow a distribution $n_f \sim \mathcal{N}(0, \gamma)$ with variance γ computed as:

$$\gamma = a \cdot \left(\frac{1}{n} \sum_{n_f=1}^{n} p_{n_f} \right)^b ,$$

rounded to the nearest integer, where a and b are tuning parameters. The trial point is accepted or rejected to minimize Equation 2.14. The full guided random walk sampling algorithm is summarized in Algorithm 2.

Algorithm 2 Guided Random Walk Sampling

Pick initial value n_f^1
Set $n_f = n_f^1$
for $k = 2 : K$ **do**
  Calculate: p_{n_f}
  Draw $s \sim \mathcal{N}(0, \gamma)$ and set $n_f^t = \text{round}(n_f + s)$
  Calculate: $p_{n_f}^t$
  if $B_k^t < B_k$ **then** Accept: Set $n_f = n_f^t, n_f^k = n_f$
  else
    Calculate $\lambda = \min\left\{\frac{B_k}{B_k^t}, 1\right\}$
    Draw $u \sim \mathcal{U}[0, 1]$
    if $u \leq \lambda$ **then** Accept: Set $n_f = n_f^t$, $n_f^k = n_f$
    else Reject: Set $n_f^k = n_f$
    end if
  end if
  Generate and record network state:
    $x(k)$ such that $(\sum_{i=1}^{n} (x(k)_i == 0)) = n_f^k$
  Compute and record network status: $y(k)_{st} = f(x(k), s, t)$
  Update p_{n_f}
end for

2.7 Estimating component failure with fragility models

Depending on the infrastructure network type, each component has a different robustness to disaster events. In this section, only pipeline failures under earthquake loadings are considered. This can be expanded to different loading and network types by using the appropriate attenuation and fragility models.

The intensity of an earthquake dissipates as it travels through the ground. The rate of dissipation depends on several parameters including geographic properties. The exact ground motion at each point being considered is determined using an *attenuation* law. For this study, the attenuation relation seen below

$$\ln Y_i(T_n) = f(M, R_i, \lambda_i, T_n) + \eta(T_n) + \epsilon_i(T_n) \tag{2.15}$$

obtained from Goda and Hong [35] is used. Here, $Y_i(T_n)$ is the ground motion intensity at location i caused by an earthquake determined in terms of the natural period, T_n. The function $f(M, R_i, \lambda_i, T_n)$ describes the attenuation of the ground motion intensity based on the earthquake magnitude M, the distance from the source to the i^{th} location, R_i, and explanatory variables λ_i. The interevent residual is $\eta(T_n)$, and $\epsilon_i(T_n)$ is the intraevent residual. These are both assumed to be Gaussian random variables with mean zero and standard deviations σ_η and σ_i, respectively.

The attenuation relation, obtained from Wang and Takada [36], is shown below

$$\log_{10} PGV = 0.725M_j + 0.00318H - 1.918 \log_{10} \left(D + 0.334e^{0.653M_j}\right) - 0.519 \tag{2.16}$$

and is used to determine the *peak ground velocity* (PGV) which can be used

to estimate the fragility of pipeline elements. Here PGV is the peak ground velocity at the site of interest in cm/s, M_j is the Japanese Meteorological Agency earthquake magnitude, H is the source depth in km, and D is the distance between the fault plane and the site in km.

Using the calculated PGV at the midpoint of the m^{th} pipeline segment, it is possible to calculate probability of pipeline failure, $p_{f,m}$ based on the HAZUS-MH fragility model [37]:

$$p_{f,m} = 1 - e^{-\nu_m \cdot \Delta l_m}. \tag{2.17}$$

Here Δl_m is the length of the m^{th} pipeline, and ν_m is the occurrence rate model described in [37], and computed as

$$\nu_m = \kappa \cdot (PVG_m)^{\tau} \tag{2.18}$$

where κ and τ are model parameters, and PVG_m is the peak ground velocity at the midpoint of the m^{th} pipeline section.

2.8 Extension to cluster connectivity

For many applications it may be desirable to know whether a set of nodes is connected with another set of nodes. For example, it may be beneficial to know whether a city, or portion of a city, is likely to have lost access to water following an earthquake. For this, more general cluster connectivity analysis is needed to determine the probability that one cluster of nodes is connected to another. Clustering of infrastructure networks has also been proposed to understand network hierarchy [38] and to facilitate analysis of large networks on multiple scales [39]. Common structure for large infrastructure networks

is assumed. Specifically, it is assumed that there are a number of densely connected components (e.g., an urban infrastructure grid), which are loosely connected to other densely connected components (e.g., another urban area) via long range links. These network structures lend themselves well to cluster connectivity analysis.

Spectral graph clustering (SGC) [40] is used to partition graphs into densely connected clusters. To split a graph into two clusters the graph Laplacian L is computed as

$$L = D - A, \tag{2.19}$$

where D is a diagonal matrix containing the degree of each node on the diagonal, and A is the adjacency matrix of the graph. Clustering is done by finding the second smallest eigenvalue of L and assigning node labels, q_i according to

$$q_i = \begin{cases} 0 & \text{if } v_i < 0, \\ 1 & \text{otherwise,} \end{cases} \tag{2.20}$$

where v_i is the i^{th} element of the eigenvector associated with the second smallest eigenvalue of L. A cluster is the set of all nodes that have the same node label q_i.

For an arbitrary k number of clusters, the first k eigenvectors of L are clustered using *k-means* clustering and used to determine the cluster assignment [40].

To determine the probability of the network failing with respect to two node clusters, the nodes are first clustered using the spectral methods described above. Two phantom nodes are introduced into the network, one for the source node, and another as the terminal node. Each node in the

source cluster is connected to the source phantom node, and each node in the terminal cluster is connected to the terminal phantom node.

CHAPTER 3

NUMERICAL EXAMPLES

3.1 California gas distribution network

This thesis uses the example of a simplified version of the California gas distribution network to numerically demonstrate the methods developed, and compare their performance. The network topology shown in Figure 3.1 is obtained from Lim et al. [39] and consists of 244 components (70 nodes and 87 bi-directional edges). In this network, each node represents a substation, and each edge represents a gas pipeline. The source and terminal nodes used in this example are indicated in Figure 3.1 with a red and blue circle, respectively.

The California gas distribution network is selected since it provides a realistic example network for which a surrogate model can be learned in a reasonable amount of time for experimentation. Furthermore, the network is large enough that direct computation of the network failure probability is infeasible. The source code for all methods developed in this thesis and experiments conducted, as well as the network topology can be accessed online from [41].

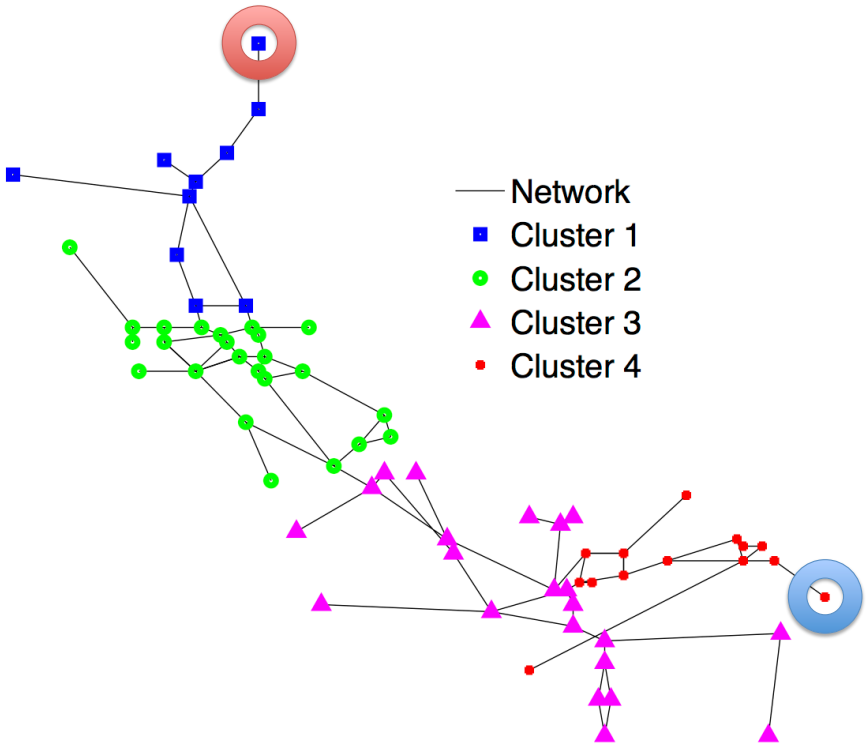


Figure 3.1: California gas distribution network with four clusters.

3.2 Comparison to AdaBoost model

In order to compare the logistic regression surrogate model against the AdaBoost based surrogate model, both models are tested on a test case for a uniform edge failure probability of $p_f = 0.15$, and is computed for $K = 5,000$ MCS. The results in Figure 3.2 show that the AdaBoost model produces a significantly higher error than the logistic regression surrogate model when compared to the MCS estimate for network failure probability. Also, the AdaBoost model requires an order of magnitude more computation time than the logistic regression surrogate model. These results are summarized in Table 3.1.

In light of these results, it is apparent that logistic regression is a more suitable choice of machine learning method to construct a surrogate model for

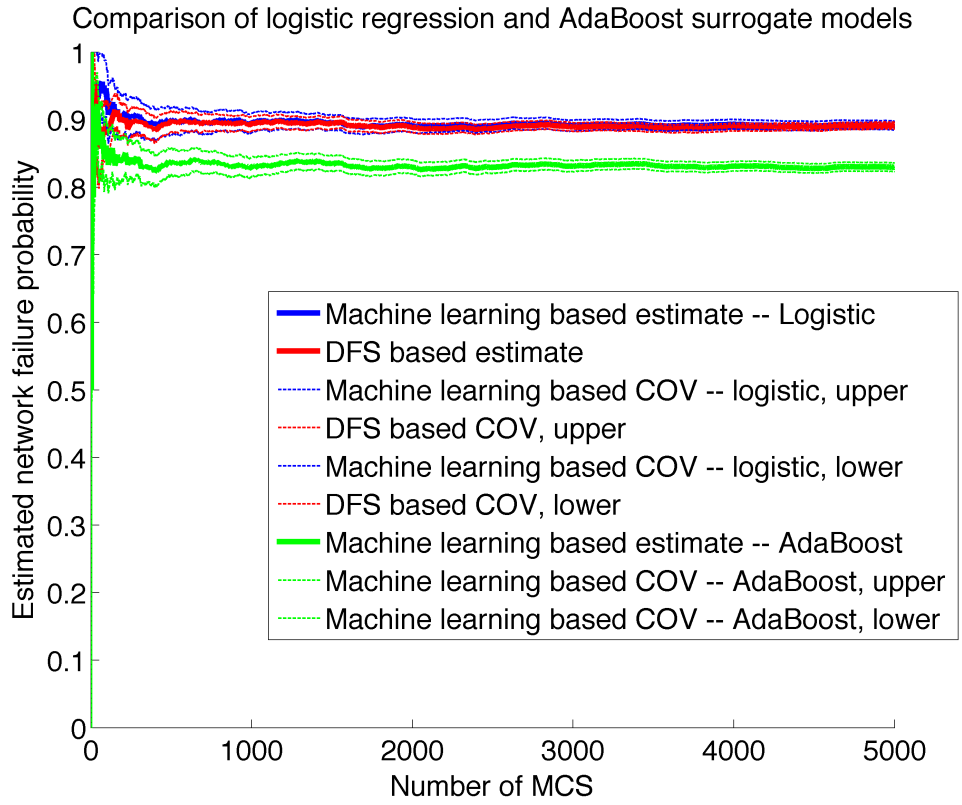


Figure 3.2: Comparison of AdaBoost and logistic regression surrogate models.

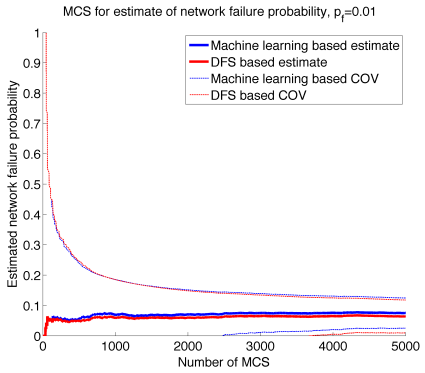
Method	Discrepancy (%)	Time (ms)
DFS	0.00	626.5
AdaBoost	6.16	33.8
Logistic	0.02	2.80

Table 3.1: Summary of results comparing AdaBoost and logistic regression-based surrogate models on California gas distribution network.

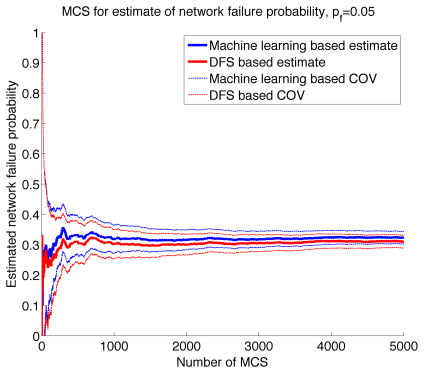
individual node connectivity. This is in line with the binary nature of node failures. While the results from the AdaBoost model produce reasonably low error, the results obtained from logistic regression are able to predict the probability of network failure with less error.

3.3 Uncorrelated edge failures

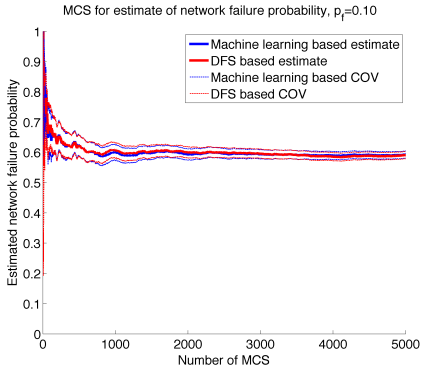
In order to test the logistic regression classifier for MCS, first the most fundamental case is considered, where only edges are allowed to fail and all edges are assumed to have the same failure probability. The failure probability for each edge is varied from $p_f = 0.05$ to $p_f = 0.45$ in different trials. The network failure probability with respect to s and t is computed using MCS with $K = 500$ samples. Both the DFS and the logistic regression surrogate model are used to determine network connectivity for each realization, and compare the estimates obtained, as well as the convergence time for each method. The results using a logistic regression classifier are shown in Table 3.2, where T_{ML} is the convergence time using the machine learning based surrogate model and T_{DFS} is the convergence time using DFS. Based on the results, the surrogate model generates the highest error for an edge failure probability of $p_f = 0.20$ for which the estimated overall network failure probability is $\hat{\Pr}(fail) = 0.8940$. The convergence results for the first five estimates are shown in Figures 3.6a–3.6f. Based on these figures, convergence occurs rapidly, with most estimates in close agreement within the first 1000 MCS.



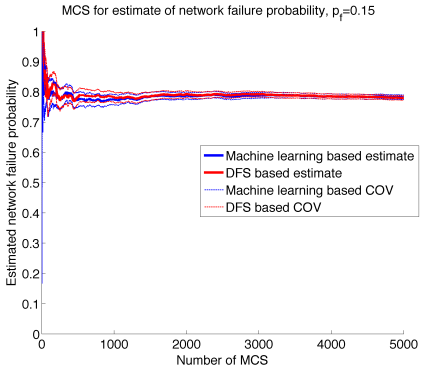
(a) Disconnection probability estimate for $p_f = 0.01$, and COV.



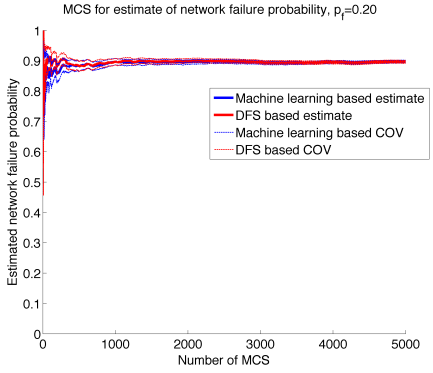
(b) Disconnection probability estimate for $p_f = 0.05$, and COV.



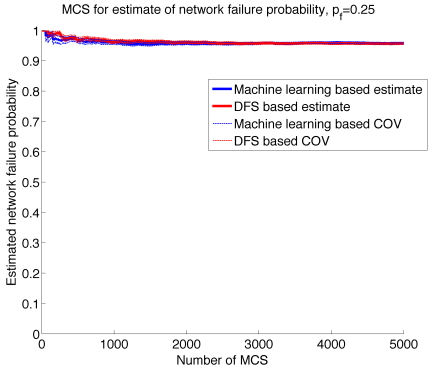
(c) Disconnection probability estimate for $p_f = 0.10$, and COV.



(d) Disconnection probability estimate for $p_f = 0.15$, and COV.



(e) Disconnection probability estimate for $p_f = 0.20$, and COV.



(f) Disconnection probability estimate for $p_f = 0.25$, and COV.

Figure 3.3: Convergence of disconnection probability estimate for different edge failure probabilities.

p_f	$\hat{\Pr}(fail)$	Logistic regression discrepancy (%)	T_{ML} (ms)	T_{DFS} (ms)
0.01	0.0638	0.0112	38.5	659.0
0.05	0.3102	0.0132	2.90	659.9
0.10	0.5906	0.0022	3.00	657.1
0.15	0.7806	0.0016	39.4	622.4
0.20	0.8964	0.0004	37.7	619.0
0.25	0.9566	0.0016	3.30	615.6

Table 3.2: Results comparing DFS and ML method for different values of uniform edge failure probability.

3.4 Correlated edge failures

In this case, nodes are not allowed to fail, and only edges are considered to be able to fail. This assumption can be relaxed by allowing the nodes to fail as well. Using the attenuation law in Section 2.7, the correlation between the failure of pipeline i and pipeline j as a function of the distance Δ, $\rho_{\ln Y_i \ln Y_j}(\Delta)$ is computed as seen in Equation 3.1 where $\rho_{\epsilon_i \epsilon_j}(\Delta)$ is the intraevent correlation, and all other terms are as defined previously.

$$\rho_{\ln Y_i \ln Y_j}(\Delta) = \frac{\sigma_\eta^2}{(\sigma_\eta + \sigma_\epsilon)^2} + \rho_{\epsilon_i \epsilon_j}(\Delta) \frac{\sigma_\epsilon^2}{(\sigma_\eta + \sigma_\epsilon)^2} \tag{3.1}$$

The PGV is computed at the midpoint of each edge using the attenuation law described in Section 2.7 assuming a $M = 7.7$ earthquake with epicenter $10km$ below the surface located at the point indicated in Figure 3.1. The length of the pipeline and the PGV are used to compute the probability of edge failure for each edge based on the location of the midpoint of each edge using (2.18) and (2.17) where $\kappa = 2$ and $\tau = 0.5$.

The probability of network failure is estimated using MCS with both DFS and the logistic regression surrogate model. The convergence results are shown in Figure 3.4. $K = 500$ samples are used, and a converged estimate with 6.4% error is obtained using the machine-learning based method.

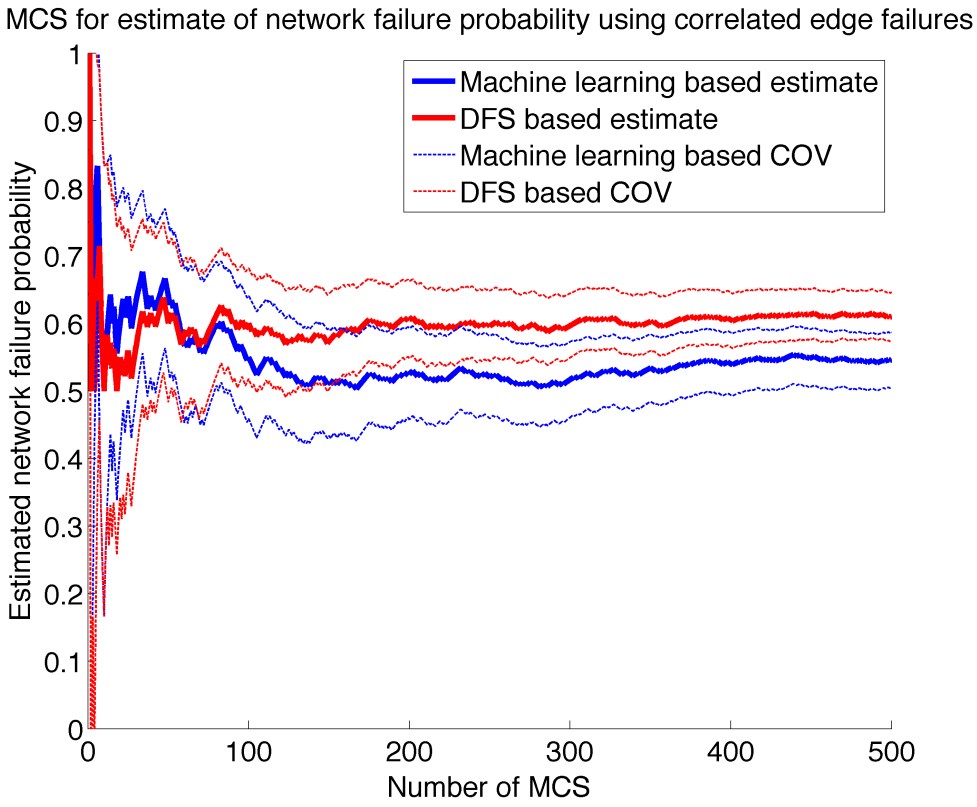


Figure 3.4: Convergence of network failure probability using correlated edges, with a $M_j = 7.7$ magnitude earthquake, attenuation law, and fragility model outlined in Section 2.7.

3.5 Cluster connectivity

To analyze cluster connectivity, the network is partitioned into relevant clusters as shown in Figure 3.1 using spectral graph clustering. In order to check connectivity between two specific clusters, phantom nodes are introduced for each cluster, which are connected to each node in their respective clusters as shown in Figure 3.5. The links from the phantom nodes to each node in the corresponding cluster are considered to be invincible, meaning that they are not allowed to fail in the simulation. This allows for quick determination of connectivity between the clusters and estimation of network failure probability by checking connectivity between the phantom nodes.

An AdaBoost surrogate model is learned for each cluster pair. The network failure probability is estimated using the California gas distribution network with uncorrelated uniform random edge failure probabilities of $p_f = 0.15$. For this simulation, $K = 5,000$ samples are drawn for the MCS in order to obtain a converged estimate.

The results shown in Table 3.3 as well as the convergence plots shown in 3.6 indicate that the machine-learning based model is able to learn the behavior of the network, and predict cluster disconnections with little error. The further the clusters are spatially separated in the graph, the more error-prone the machine-learning based method is. For clusters in close proximity, the machine-learning based model is able to predict disconnections with perfect accuracy and in less time than the DFS-based method. That is because for these networks, as shown in Figure 3.5, some cluster pairs are connected with only a small number of edges. Therefore, the machine-learning based surrogate model is able to learn a very simple rule for determining whether or not these clusters are connected.

Cluster pair	$\hat{\Pr}(fail)$	Discrepancy (%)	T_{ML} (ms)	T_{DFS} (ms)
1 - 2	0.0204	0.00	7.10	720.7
1 - 3	0.2132	4.04	14.8	671.8
1 - 4	0.3160	6.92	17.1	650.8
2 - 3	0.1502	0.00	3.60	709.3
2 - 4	0.2734	4.16	14.4	697.2
3 - 4	0.0286	0.00	2.70	690.7

Table 3.3: Summary of results for all six cluster-to-cluster models with edge failure probability $p_f = 0.15$ using an AdaBoost surrogate model. Computation times for machine learning method T_{ML} and depth first search method T_{DFS} provided.

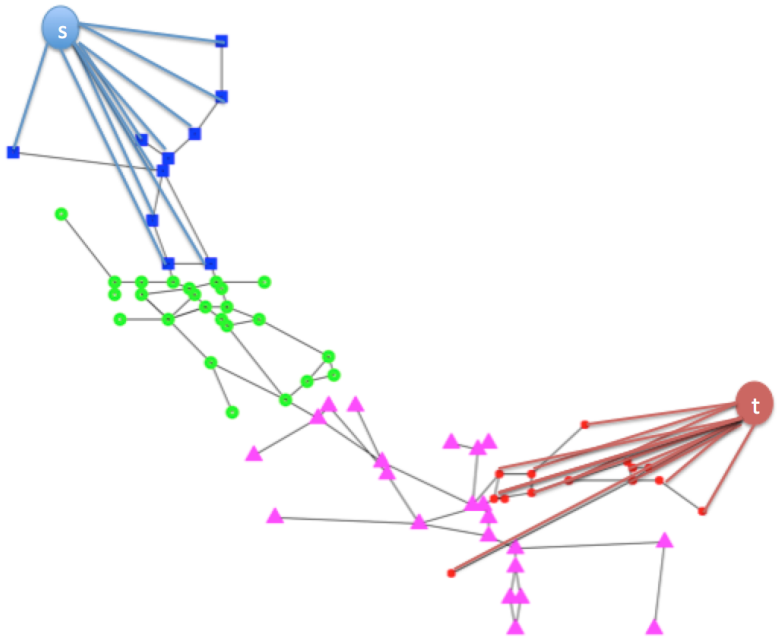
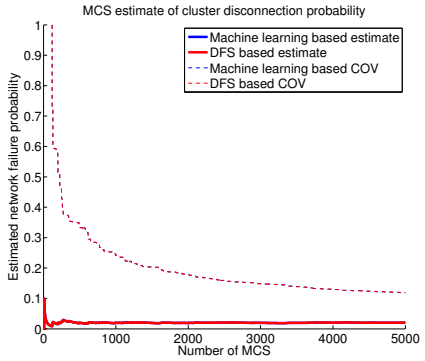
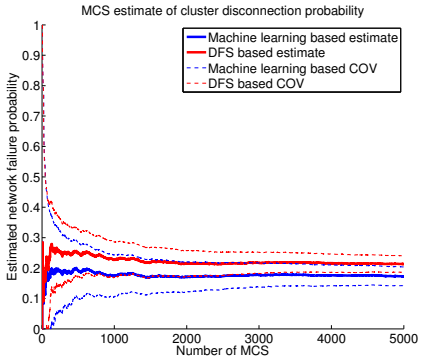


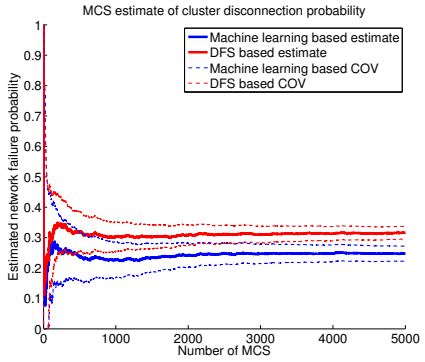
Figure 3.5: Graphical representation of how phantom nodes are introduced for accelerated cluster computations. Specific example showing phantom nodes introduced to check connectivity between clusters 1 and 4.



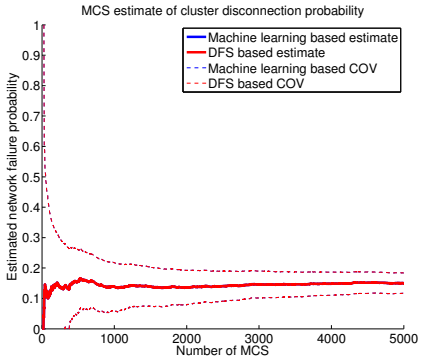
(a) Cluster disconnection probability estimate for clusters 1 and 2, $p_f = 0.15$.



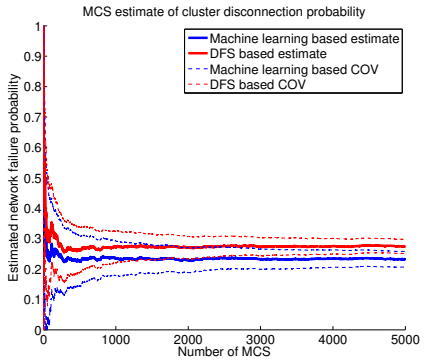
(b) Cluster disconnection probability estimate for clusters 1 and 3, $p_f = 0.15$.



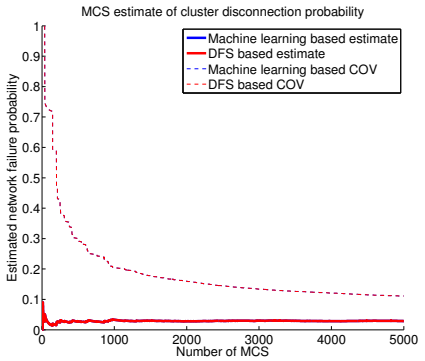
(c) Cluster disconnection probability estimate for clusters 1 and 4, $p_f = 0.15$.



(d) Cluster disconnection probability estimate for clusters 2 and 3, $p_f = 0.15$.



(e) Cluster disconnection probability estimate for clusters 2 and 4, $p_f = 0.15$.



(f) Cluster disconnection probability estimate for clusters 3 and 4, $p_f = 0.15$.

Figure 3.6: Convergence of cluster disconnection probability estimate for different all six cluster pairs using $p_f = 0.15$.

CHAPTER 4

CONCLUSIONS AND FUTURE WORK

The methods and results presented in this thesis indicate that it is feasible and practical to use machine-learning-based surrogate in place of exact connectivity checks when conducting MCS to estimate the probability of network failure. These results are significant in the reliability community since they demonstrate that machine learning methods can be used in place of exact connectivity checks to greatly improve the speed of convergence of MCS estimates on network failure probability for stochastic networks.

Specifically, the numerical examples show that the best performance on a graph with uniform random edge failure is obtained when using a logistic regression-based surrogate model. Similar convergence is achieved when using correlated edge failures. However, these results are obtained with greater error than the results from MCS with uncorrelated uniform edge failure probability.

Based on the results presented, other machine learning methods can be applied to learn network structure and construct a surrogate model for network failure. For example, the AdaBoost classifier is used to train a surrogate model. This model is found to be less accurate than the logistic regression model when computing the probability of network failure.

The methods developed in this thesis are also shown to be applicable to extensions to the two-terminal reliability problem such as a cluster-reliability problem. In this case, different AdaBoost-based surrogate models are learned

for each cluster pair of interest. It is observed that the geographically closer the clusters are, the more accurately the machine-learning-based surrogate model is able to predict cluster connectivity.

The potential gains to society that real-time SRA or near-real-time SRA can provide are immense. Further research will help make this goal possible. While the results in this thesis provide one method to improve the convergence speed of network failure probability, other methods must also be investigated, and possibly combined to form a comprehensive framework for quickly estimating the probability of connectivity of network components following a natural disaster.

One extension to this specific work, which may prove beneficial to the community, is the development of surrogate models that can be used in conjunction with MCS to estimate the probability of network disconnections for the k-terminal reliability, or even the all-terminal reliability problem. This may prove tricky, since significantly more information about the network must be learned by the classifier in order to make such predictions. Solving this problem would be very beneficial to the community, since it would provide a fast and accurate method for evaluating the probability of disconnection between any two arbitrary nodes in the network.

Another possible extension to this work, which may improve the quality of the predictions is to use some variable selection algorithm such as *Bayesian information criteria* (BIC) to select only the significant parameters of the model [42]. This would utilize the same framework as presented in this thesis, but would rely on a different surrogate model. The benefit of using variable selection being that a more parsimonious model may be less prone to over-fitting, and could require less computation time for classification. Such improvements are in line with the general work presented in this thesis.

REFERENCES

[1] M. Bruneau, S. E. Chang, R. T. Eguchi, G. C. Lee, T. D. O'Rourke, A. Reinhorn, M. Shinozuka, K. Tierney, W. A. Wallace, and D. von Winterfeldt, "A framework to quantitatively assess and enhance the seismic resilience of communities," *Earthquake Spectra*, vol. 19, no. 4, pp. 733–752, 2003.

[2] A. Boin and A. McConnell, "Preparing for critical infrastructure breakdowns: The limits of crisis management and the need for resilience," *Journal of Contingencies and Crisis Management*, vol. 15, no. 1, pp. 50–59, 2007.

[3] J. Song and S.-Y. Ok, "Multi-scale system reliability analysis of lifeline networks under earthquake hazards," *Earthquake Engineering and Structural Dynamics*, vol. 39, pp. 259–279, 2009.

[4] E. D. Vugrin, D. E. Warren, M. A. Ehlen, and R. C. Camphouse, "A framework for assessing the resilience of infrastructure and economic systems," in *Sustainable and Resilient Critical Infrastructure Systems*, K. Gopalakrishnan and S. Peeta, Eds. Springer Berlin Heidelberg, 2010, pp. 77–116.

[5] D. Reed, K. Kapur, and R. Christie, "Methodology for assessing the resilience of networked infrastructure," *Systems Journal, IEEE*, vol. 3, no. 2, pp. 174–180, June 2009.

[6] A. Der Kiureghian and J. Song, "Multi-scale reliability analysis and updating of complex systems by use of linear programming," *Reliability Engineering & System Safety*, vol. 93, no. 2, pp. 288–297, 2008.

[7] FEMA, "Crisis Response and Disaster Resilience 2030: Forging Strategic Action in an Age of Uncertainty," Federal Emergency Management Agency, Tech. Rep., 2012.

[8] DHS, "National Infrastructure Protection Plan," Department of Homeland Security, Tech. Rep., 2013.

[9] M. O. Ball, "Computational complexity of network reliability analysis: An overview," *IEEE Transactions on Reliability*, vol. 35, no. 3, pp. 230–239, 1986.

[10] S. Soh and S. Rai, "An efficient cutset approach for evaluating communication-network reliability with heterogeneous link-capacities," *IEEE Transactions on Reliability*, vol. 54, no. 1, pp. 133–144, 2005.

[11] J. L. Cook and J. E. Ramirez-Marquez, "Two-terminal reliability analyses for a mobile ad hoc wireless network," *Reliability Engineering System Safety*, vol. 92, no. 6, pp. 821 – 829, 2007.

[12] S. Satitsatian and K. Kapur, "An algorithm for lower reliability bounds of multistate two-terminal networks," *IEEE Transactions on Reliability*, vol. 55, no. 2, pp. 199–206, June 2006.

[13] J. S. Provan and M. O. Ball, "The complexity of counting cuts and of computing the probability that a graph is connected," *SIAM Journal on Computation*, vol. 12, no. 4, pp. 777–788, 1983.

[14] A. Rosenthal, "Computing the reliability of complex networks," *SIAM Journal on Applied Mathematics*, vol. 32, no. 2, pp. 384–393, 1977.

[15] C. Lucet and J.-F. Manouvrier, "Exact methods to compute network reliability," in *Statistical and Probabilistic Models in Reliability*. Springer, 1999, pp. 279–294.

[16] D. R. Karger, "A randomized fully polynomial time approximation scheme for the all-terminal network reliability problem," *SIAM Review*, vol. 43, no. 3, pp. 499–522, 2001.

[17] H. H. M. Hwang, H. Lin, and M. Shinozuka, "Seismic performance assessment of water delivery systems," *Journal of Infrastructure Systems*, vol. 4, no. 3, pp. 118–125, 1998.

[18] G. Rubino and B. Tuffin, *Rare Event Simulation using Monte Carlo Methods*. Chichester, UK: John Wiley & Sons, 2009.

[19] R. Tarjan, "Depth-first search and linear graph algorithms," *SIAM Journal on Computing*, vol. 1, no. 2, pp. 146–160, 1972.

[20] Z. Qian, C. C. Seepersad, V. R. Joseph, J. K. Allen, and C. F. J. Wu, "Building surrogate models based on detailed and approximate simulations," *Journal of Mechanical Design*, vol. 128, pp. 668–677, 2005.

[21] X. Wan, J. F. Pekny, and G. V. Reklaitis, "Simulation-based optimization with surrogate modelsapplication to supply chain management," *Computers & Chemical Engineering*, 2003.

[22] N. P. Hummon and P. Doreian, "Computational methods for social network analysis," *Social Networks*, vol. 12, no. 4, pp. 273 – 288, 1990.

[23] S. Even and R. E. Tarjan, "Network flow and testing graph connectivity," *SIAM Journal on Computing*, vol. 4, no. 4, pp. 507–518, 1975.

[24] E. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, 1956.

[25] J. Hopcroft and R. Tarjan, "Efficient planarity testing," *Journal of the Association for Computing Machinery*, vol. 21, no. 4, pp. 549–568, 1974.

[26] T. Jansson, L. Nilsson, and M. Redhe, "Using surrogate models and response surfaces in structural optimization with application to crashworthiness design and sheet metal forming," *Structural and Multidisciplinary Optimization*, vol. 25, no. 2, pp. 129–140, 2003.

[27] J. Meirlaen, B. Huyghebaert, F. Sforzi, L. Benedetti, and P. Vanrolleghem, "Fast, simultaneous simulation of the integrated urban wastewater system using mechanistic surrogate models," *Water Science and Technology*, vol. 43, no. 7, pp. 301–307, 2001.

[28] X. Wan, J. F. Pekny, and G. V. Reklaitis, "Simulation-based optimization with surrogate modelsapplication to supply chain management," *Computers & Chemical Engineering*, vol. 29, no. 6, pp. 1317–1328, 2005.

[29] S. K. Palei and S. K. Das, "Logistic regression model for prediction of roof fall risks in bord and pillar workings in coal mines: An approach," *Safety Science*, vol. 47, no. 1, pp. 88 – 96, 2009.

[30] C.-J. Lin, R. C. Weng, and S. S. Keerthi, "Trust region newton method for logistic regression," *The Journal of Machine Learning Research*, vol. 9, pp. 627–650, 2008.

[31] Y. Freund and R. E. Schapire, "A short introduction to boosting," *Journal of Japanese Society for Artificial Intelligence*, vol. 14, no. 5, pp. 771–780, 1999.

[32] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee, "Boosting the margin: a new explanation for the effectiveness of voting methods," *The Annals of Statistics*, vol. 26, no. 5, pp. 1651–1686, 1998.

[33] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intelligent Data Analysis*, vol. 6, no. 5, pp. 429–450, 2002.

[34] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[35] K. Goda and H. P. Hong, "Spatial correlation of peak ground motions and response spectra," *Bulletin of the Seismological Society of America*, vol. 98, no. 1, pp. 354–365, 2008.

[36] M. Wang and T. Takada, "Macrospatial correlation model of seismic ground motions," *Earthquake Spectra*, vol. 21, no. 4, p. 11371156, 2005.

[37] FEMA, *HAZUS MH-MR3 Technical Manual*. Washington, DC: Federal Emergency Management Agency, 2008.

[38] C. Gmez, M. Sanchez-Silva, L. Dueas-Osorio, and D. Rosowsky, "Hierarchical infrastructure network representation methods for risk-based decision-making," *Structure and Infrastructure Engineering*, vol. 9, no. 3, pp. 260–274, 2013.

[39] H.-W. Lim, J. Song, and K. Nolan, "Seismic reliability assessment of lifeline networks using cluster-based multi-scale approach," *Earthquake Engineering Structural Dynamics*, in print, doi:10.1002/eqe.2472.

[40] U. von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, 2007.

[41] R. Stern, "California gas distribution network source code," 2015. [Online]. Available: https://github.com/Lab-Worl/systemreliability

[42] G. Schwarz, "Estimating the dimension of a model," *The Annals of Statistics*, vol. 6, no. 2, pp. 461–464, 1978.